Observations on COTS Software Integration Effort Based on the COCOTS Calibration Database

Chris Abts, M.S.
University of Southern California
Salvatori Hall Room 328
941 W. 37th Place
Los Angeles, CA 90089 USA
+1 213 740 6470
cabts@sunset.usc.edu

Barry W. Boehm, Ph.D.
University of Southern California
Salvatori Hall Room 328
941 W. 37th Place
Los Angeles, CA 90089 USA
+1 213 740 5703
boehm@sunset.usc.edu

Elizabeth Bailey Clark, Ph.D.
Software Metrics, Inc.

4345 High Ridge Road
Haymarket, VA 20169 USA
+1 703 754 0115
BetsyClark@erols.com

ABSTRACT
As the use of commercial-off-the-shelf (COTS) components becomes ever more prevalent in the creation of large software systems, the need for the ability to reasonably predict the true lifetime cost of using such software components grows accordingly. This paper presents empirically-based findings about the effort associated with activities found to be significant in the development of systems using COTS components. The findings are based upon data collected for the purpose of calibrating the COCOTS [1,2] COTS software integration cost model, an extension to the COCOMO II [3] cost model designed to capture costs COCOMO does not. A brief overview of COCOTS is presented to put the data in perspective, including its relation to COCOMO II. A set of histograms is then shown summarizing the effort data collected to date, then the latest calibration results. The paper concludes with some observations suggested by an examination of that calibration data.

Keywords
COTS, COTS integration, COTS assessment, COTS software lifecycle, COCOMO II, cost estimation, effort and schedule estimation, metrics, software engineering

1   INTRODUCTION
COCOTS is the acronym for the *COnstructive COTS* integration cost model, where COTS in turn is short for *commercial-off-the-shelf,* and refers to those pre-built, commercially available software components that are becoming ever more important in the creation of new software systems.

The rationale for building COTS-containing systems is that they will involve less development time by taking advantage of existing, market proven, vendor supported products, thereby reducing overall system development costs. But there are two defining characteristics of COTS software, and they drive the whole COTS usage process:

1) the COTS product source code is not available to the application developer, and

2) the future evolution of the COTS product is not under the control of the application developer.

(Note: in some cases, COTS software does come as source code, but for the purposes of our model, if the developer does anything to that code other than compile it unchanged into his own code, we treat that as a *reuse* item and model its usage as *adapted* code within COCOMO II itself.)

Because of these characteristics, there is a trade-off in using the COTS approach in that new software development time can indeed be reduced, but generally at the cost of an increase in software component integration work. The long term cost implications of adopting the COTS approach are even more profound, because you are in fact adopting a new way of doing business from the moment you start considering COTS components for your new system to the day you finally retire that system. This is because COTS software is not static; it continually evolves in response to the market, and you as the system developer must adopt methodologies that cost-effectively manage the use of those evolving components.

2   RELATION TO COCOMO II
The software development cost estimation model COCOMO (*COnstructive COst MOdel*) was originally published in 1981 [4], and has recently been reincarnated as COCOMO II [5] to reflect current software development practice. It creates effort and schedule estimates for software systems built using a variety of techniques or approaches. The first and primary approach modeled by COCOMO is the use of system components that are built from scratch, that is, *new* code. But COCOMO II also allows you to model the case in which system components are built out of pre-existing source code that is modified or adapted to your current purpose, i.e., *reuse* code. The key word in the preceding sentence is *source*. Even though you

are not building the reuse component from scratch, you still have access to the component's source code and can rewrite or modify it specifically to suit your needs.

What COCOMO II currently does not model is that case in which you do not have access to a pre-existing component's source code. You have to take the component as is, working only with its executable file, and at most are able to build a software shell *around* the component to adapt its functionality to your needs.

This is where COCOTS comes in. COCOTS is being designed specifically to model the unique conditions and practices highlighted in the preceding section that obtain when you incorporate COTS components into the design of your larger system.

3   COCOTS MODEL OVERVIEW

COCOTS at the moment is composed of four related submodels, each addressing individually what we have identified as the four primary sources of COTS software integration costs. (This is a key point. COCOTS currently deals only with *initial* integration efforts. We have already begun taking the first steps toward expanding the model to cover long-term lifecycle and maintenance costs associated with using COTS components. However, that topic will remain outside the scope of this paper.)

Initial integration costs are due to the effort needed to perform (1) candidate COTS component assessment, (2) COTS component tailoring, (3) the development and testing of any integration or "glue" code (sometimes called "glueware" or "binding" code) needed to plug a COTS component into a larger system, and (4) increased system level programming and testing due to volatility in incorporated COTS components.

(A fifth cost source was actually identified early in our research. This was the cost related to the increased verification and validation effort unrelated to COTS volatility but still usually required when using COTS components. But attempts have been made to capture these costs within the glue code and tailoring submodels directly rather than specify a fifth independent submodel.)

Assessment is the process by which COTS components are selected for use in the larger system being developed. *Tailoring* refers to those activities that would have to be performed to prepare a particular COTS program for use, regardless of the system into which it is being incorporated, or even if operating as a stand-alone item. These are things such as initializing parameter values, specifying I/O screens or report formats, setting up security protocols, etc. *Glue code* development and testing refers to the new code external to the COTS component itself that must be written in order to plug the component into the larger system. This code by nature is unique to the particular context in which the COTS component is being used, and must not be confused with tailoring activity as defined above. *Volatility*

in this context refers to the frequency with which new versions or updates of the COTS software being used in a larger system are released by the vendors over the course of the system's development and subsequent deployment.
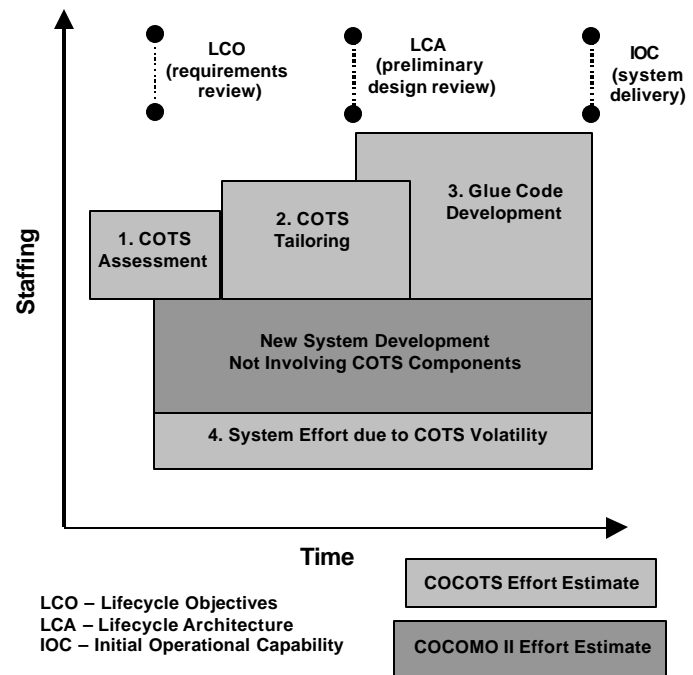

Figure 1. Sources of Effort

Figure 1 illustrates how the modeling of these effort sources in COCOTS is related to effort modeled by COCOMO II. The figure represents the total effort to build a software system out of a mix of new code and COTS components as estimated by a combination of COCOMO II and COCOTS. The central block in the diagram indicates the COCOMO II estimate, that is, the effort associated with any newly developed software in the system. . The smaller, exterior blocks indicate COCOTS estimates, that effort associated with the COTS components in the system. The relative sizes of the various blocks in this figure is a function of the number of COTS components relative to the amount of new code in the system, and of the nature of the COTS component integration efforts themselves. The more complex the tailoring and/or glue code-writing efforts, the larger these blocks will be relative to the assessment block. Also, note that addressing the system wide volatility due to volatility in the COTS components is an effort that will obtain throughout the entire system development cycle, as indicated by the long block running along the bottom of the figure.

Finally, we caution the reader to not draw the erroneous conclusion that the various sources of effort being discussed can be as cleanly parsed as would appear to be indicated by the distinct boundaries drawn for each block.

The reality is that rather than sharp lines, these boundaries would probably be more realistically shown as areas of different shading that blend into each other. This is particularly true for the boundary between the bottom volatility block and the larger system block.

4   CALIBRATION DATABASE SUMMARIES

We currently have 20 data points (information on historical industrial software projects using COTS components) in our possession, and our data collection efforts are ongoing. The histograms that follow indicate how many projects (or elements of projects) collected to date manifest a given attribute.

The distribution of data points across various project domains:

- Air Traffic Management (40%)
- Business (including databases) (15%)
- Comm/Navigation/Surveillance (20%)
- Logistics (5%)
- Mission Planning (5%)
- Operations (10%)
- Web-based Maps (5%)

The classes of COTS products found in these projects:

- Back office/retail
- Configuration mgmt/build tools
- Databases
- Data conversion packages
- Device drivers
- Disk arrays
- Compilers
- Communication protocols/packages
- Emulators
- Engineering tools
- Graphic information systems
- GUIs/GUI builders
- Middleware
- Network managers
- Operating Systems
- Problem mgmt
- Report generators
- Software process tools
- Telecommunication & infrastructure
- Telemetry analysis
- Telemetry processing
- Word processing

The preceding demonstrates that we have the beginnings of a varied cross section of domains and products, which is desirable when attempting a general calibration of an estimation model. Also, nearly all of the projects went through initial delivery within the last 3 years, so the data is recent, which is also helpful; i.e., it should reflect current practice.
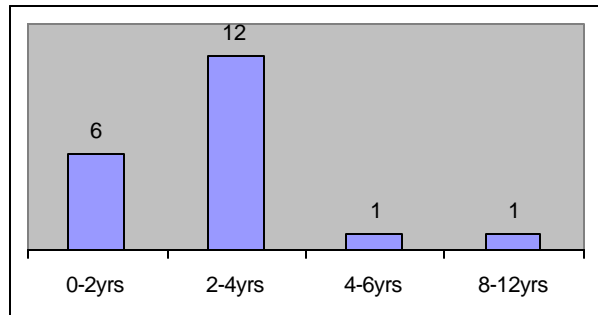
The development processes followed by the various projects while creating these COTS-based systems:

- Waterfall (40%)
- Spiral (35%)
- Incremental (20%)
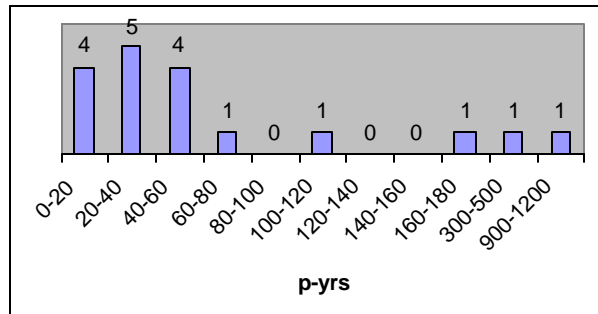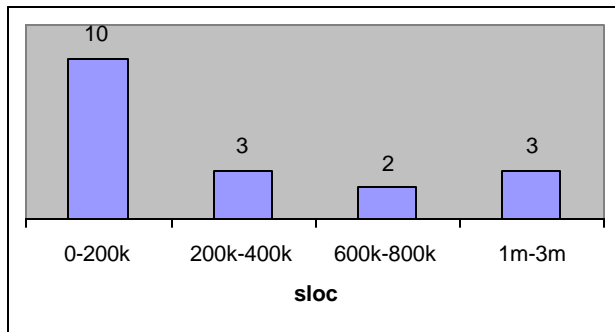- Evolution/Prototype (5%)

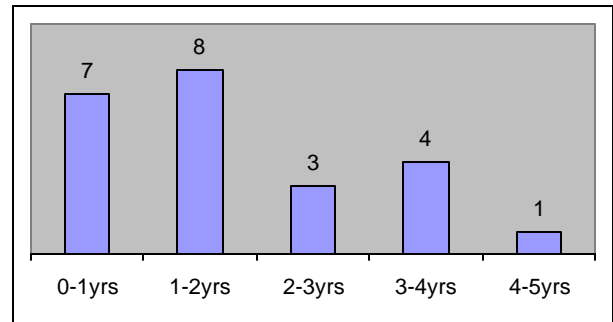Project delivery dates:
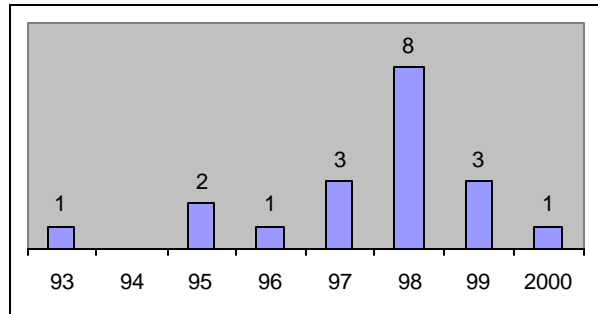
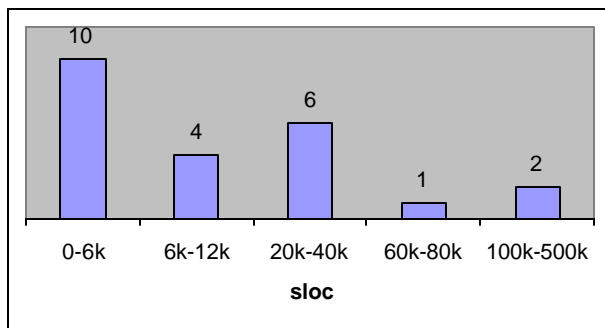Overall project duration:

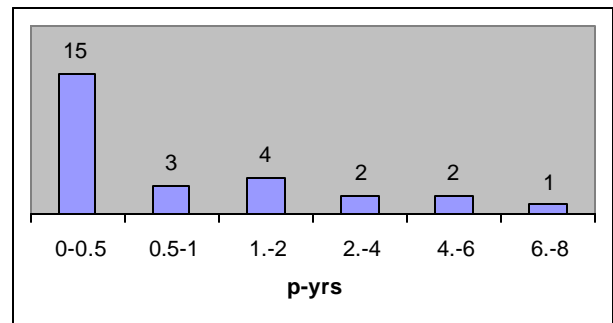Overall project effort:

Overall project SLOC:

Project glue SLOC (reflects multi-COTS classes per project):

Project COTS assessment activity duration:

Project COTS tailoring activity duration (reflects multi-COTS classes per project):

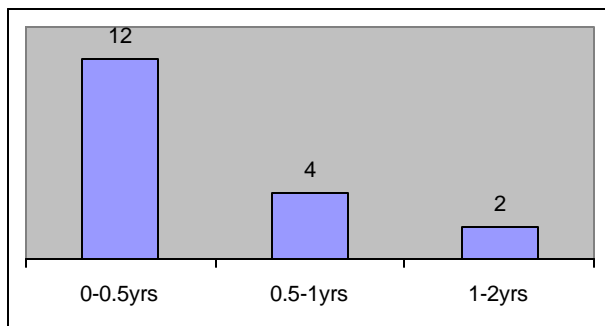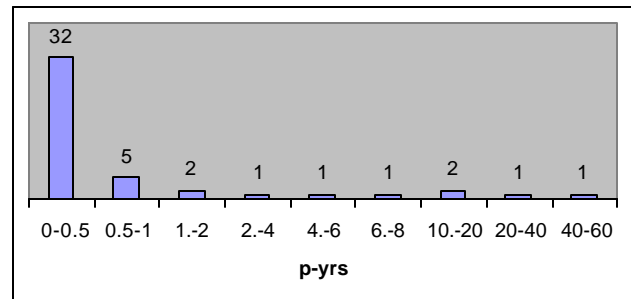Project COTS glue code creation duration (reflects multi-COTS classes per project):

Project COTS assessment activity effort (reflects multi-COTS classes per project):

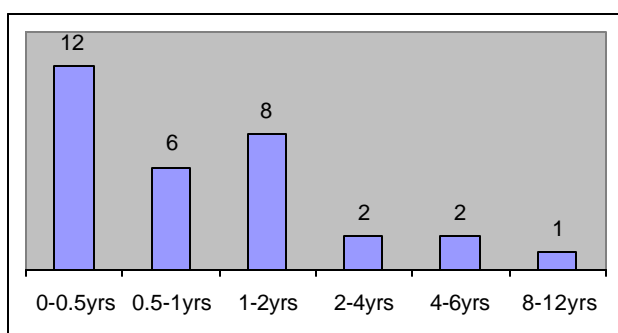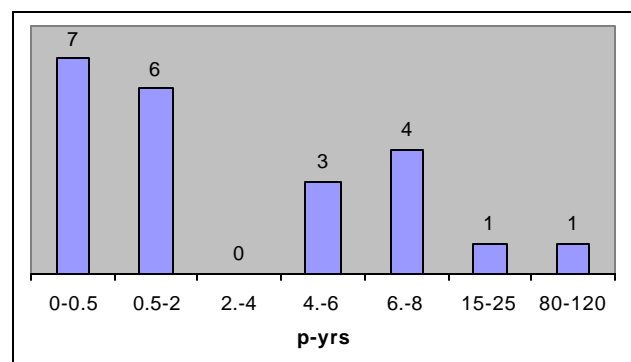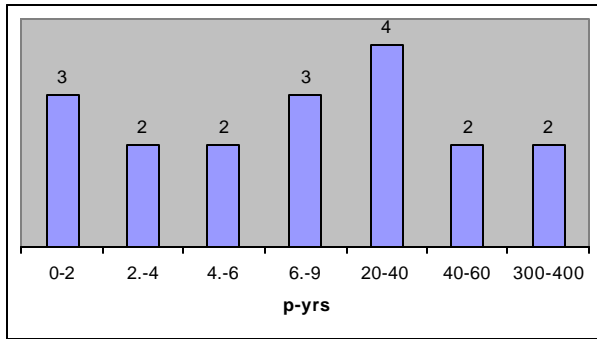Project COTS tailoring activity effort (reflects multi-COTS classes per project):
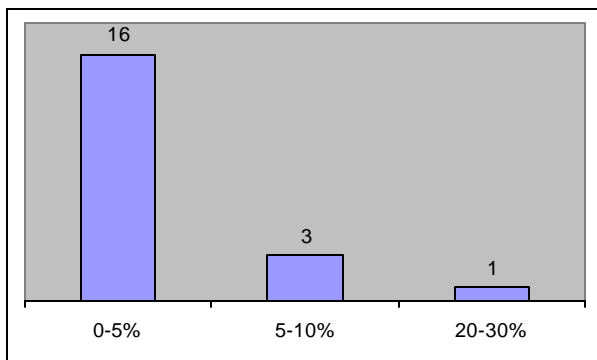

Project COTS glue coding activity effort (reflects multi-COTS classes per project):
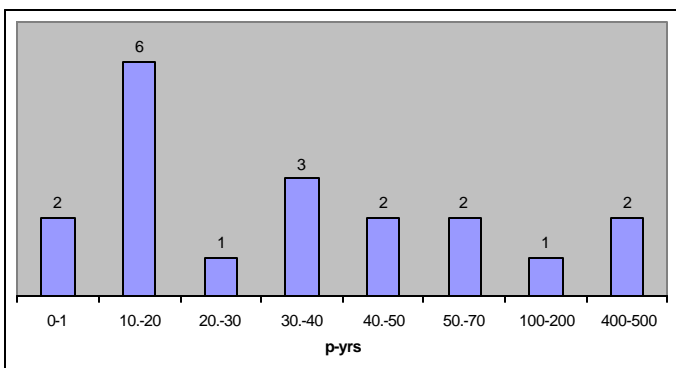
Project system effort due to COTS volatility:

Project percentage system rework effort due to COTS volatility:

Project total COTS effort in system (assessment+ tailoring+glue code+ volatility effort):

5   OBSERVATIONS

The data presented here represents an interim state of the COCOTS calibration database. As of this writing, the database contains information on some 20 industrial software projects--and data collection is continuing. As the database grows, even greater insights regarding COTS software integration will be possible. Meanwhile, there are enough data points currently available to draw some preliminary conclusions. (Keep in mind, though, that the statements that follow are at best rules of thumb gleaned from trends that seem apparent in the data—there are more than a few countervailing examples in each case.)

All COTS assessment activity on a project appears most typically to be completed within 6 calendar months, with no more than 6 person-months in effort expended to do assessment for any given class of COTS components.

The spread in typical schedule for completion of the tailoring of all COTS components is greater than that for assessment, but still, more often than not, all tailoring appears to be completed within 6 calendar months--with again, no more than 6 person-months of effort expended for any given class of COTS products.

The data indicates Glue code typically takes longer and requires more effort to complete than tailoring. This may be because the "intellectual effort" required to simply configure (or tailor) a given COTS product is usually less than that required to create code around it that is not only new but also highly constrained—the situation that exists with glue code. The typical overall schedule for creation of all glue code in a system appears to be from 1 to 2 years, with up to 2 person-years of effort expended.

The effort associated with managing the impact of the volatility of COTS components on the larger system continues throughout the life of the project. Rather than trying to determine a distinct schedule for "managing volatility," its impact on overall project schedule would probably best be accounted for by combining that effort with all other project effort and then feeding that aggregate effort into a schedule model similar to the COCOMO II schedule equation. In the meantime, the relatively low percentage of system rework due to COTS volatility reported for most of the projects in our database probably reflects the database's current reporting horizon: the end of the initial development phase. As COCOTS is expanded to cover the long-term operations & maintenance phase of a project, the relative percentage of system rework due to COTS volatility can be expected to increase.

6   LATEST CALIBRATION RESULTS

The performance of the first three submodels based on the most recent calibration efforts are presented in this section. Be aware that this is still considered an experimental, interim calibration. Because the calibration database is still relatively small, all data points available were included in the calibration and the tests of the model's accuracy were then run against those same data points. The more rigorous approach of course is to set aside a subset of the available data to serve as a control group against which to test the model's accuracy based upon a calibration using the other remaining data points. This will be done for COCOTS once a sufficient number of additional data

Total FPAE (PM)

Program	Est.	Rcpt.	%Rerr
A	110.998	87	27.58%
B	17.25	40	-56.88%
C	19.584	4	389.60%
D	45.993	69	-33.34%
G	12	7	71.43%
H	5.001	6	-16.65%
I	17.227	21	-17.97%
J	14.5	24	-39.58%
K	24.329	18	35.16%
L	7.503	7.5	0.04%
M	1.248	1.25	-0.16%
N	9.668	4.5	114.84%
Q	13.386	6	123.10%
R	29.931	147.5	-79.71%
S	68.53	58	18.16%

Total PTE (PM)

Program	Est.	Rcpt.	%Rerr
A	324.3163	268	21.01%
B	2.07	3	-31.00%
D	12.67	12	5.58%
G	8	11	-27.27%
H	21.96	26	-15.54%
J	38.01	38	0.03%
K	18.15	90	-79.83%
L	50.82	186	-72.68%
M	3	3	0.00%
N	28.15	16	75.94%
O	26	12	116.67%
P	137.79	42	228.07%
Q	4	4	0.00%
R	46.3309	640	-92.76%
S	61.3228	183	-66.49%
T	11.32	12	-5.67%
U	6	50	-88.00%

Total Glue Code (PM)

Program	Est.	Rcpt.	%Rerr
A	4.35	12	-63.75%
B	711.82	60	1086.36%
D	4.94	74	-93.32%
E	196.43	250	-21.43%
G	72.08	84	-14.19%
H	0.79	6	-86.90%
J	0.83	1	-17.27%
K	1.58	4	-60.49%
L	1.78	7	-74.64%
M	38.76	81	-52.15%
N	16.94	12	41.14%
O	595.03	1411	-57.83%
P	89.39	96	-6.89%
Q	78.50	72	9.03%
S	29.84	18	65.79%
U	122.74	75	63.65%

points are added to the calibration database to make this practical. Until then, it is felt that results from the current calibration are still useful as a means to gain insight into the model's performance.

Total COTS IE (PM)

Program	Est.	Rcpt.	%Rerr
R	76.26	787.5	-90.32%
L	60.10	200.5	-70.03%
K	44.06	112	-60.66%
D	63.61	155	-58.96%
O	621.03	1423	-56.36%
M	43.01	85.25	-49.55%
S	159.69	259	-38.34%
H	27.75	38	-26.98%
E	196.43	250	-21.43%
I	17.23	21	-17.97%
J	53.34	63	-15.34%
G	92.08	102	-9.73%
T	11.32	12	-5.67%
U	128.74	125	2.99%
Q	95.89	82	16.93%
A	439.66	367	19.80%
P	227.18	138	64.62%
N	54.75	32.5	68.48%
C	19.58	4	389.60%
B	731.14	103	609.84%

10 out of 20 (50%) of COTS Integration estimates within (+/-) 40% of actuals.

7 CONCLUSIONS

The next step in analyzing the data would be to look for correlation between various items. For example, between the overall size of a given project and the amount of glue code typically found in a COTS-based system; between overall project size and total COTS integration effort; between integration schedule & effort and individual classes of COTS components, etc.

It might also be useful to attempt calibrations of the model across various stratified subsets of the data, for example, against all projects that were considered to be successful, or against all projects of a certain size. Experience with COCOMO II has shown that basing calibrations on stratified data can greatly improve the performance of a model for projects that are homogeneous with the data stratification criteria.

REFERENCES

1. Abts, C., Boehm, B. and Bailey Clark, B., "COCOTS: a COTS software integration cost model," *Proceedings ESCOM-SCOPE 2000 Conference,* April 2000.

2. Official COCOTS website: http://sunset.usc.edu/COCOTS/cocots.html.

3. Boehm, B., Clark, B., Horowitz, E., Madachy, R., Shelby, R. and Westland, C., "Cost Models for Future Software Lifecycle Processes: COCOMO 2.0," in *Annals of Software Engineering*, 1995.

4. Boehm, B. *Software Engineering Economics*, Prentice Hall, NJ, 1981.

5. Boehm, B. et al, *Software Cost Estimation with COCOMO II*, Prentice Hall, NJ, July 2000.